



THE  
POWER  
TO KNOW.

**SAS<sup>®</sup> Enterprise Miner<sup>™</sup> and  
SAS<sup>®</sup> Text Miner Procedures  
Reference for SAS<sup>®</sup> 9.1.3  
The TREEBOOST Procedure  
(Book Excerpt)**

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., SAS® *Enterprise Miner™* and SAS® *Text Miner Procedures: Reference for SAS® 9.1.3*, Cary, NC: SAS Institute Inc.

**SAS® Enterprise Miner™ and SAS® Text Miner Procedures: Reference for SAS 9.1.3**

Copyright © 2008 by SAS Institute Inc., Cary, NC, USA.

All rights reserved. Produced in the United States of America.

**For a Web download or e-book:** Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

**U.S. Government Restricted Rights Notice.** Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

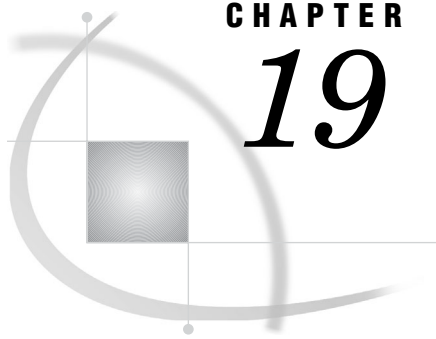
SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st electronic book, October 2008

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at [support.sas.com/publishing](http://support.sas.com/publishing) or call 1-800-727-3228.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.



## CHAPTER

## 19

## The TREEBOOST Procedure

---

<i>Overview: TREEBOOST Procedure</i>	399
<i>Syntax: TREEBOOST Procedure</i>	399
<i>PROC TREEBOOST Statement</i>	400
<i>ASSESS Statement</i>	402
<i>CODE Statement</i>	403
<i>DECISION Statement</i>	403
<i>FREQ Statement</i>	404
<i>IMPORTANCE Statement</i>	404
<i>INPUT Statement</i>	404
<i>MAKEMACRO Statement</i>	405
<i>PERFORMANCE Statement</i>	405
<i>SAVE Statement</i>	406
<i>SCORE Statement</i>	406
<i>SUBSERIES Statement</i>	407
<i>TARGET Statement</i>	407
<i>Details: TREEBOOST Procedure</i>	407
<i>FIT= Output Data Set</i>	407

---

### Overview: TREEBOOST Procedure

Tree boosting creates a series of decision trees that together form a single predictive model. A tree in the series is fit to the residual of the prediction from the earlier trees in the series. The residual is defined in terms of the derivative of a loss function. For squared error loss with an interval target, the residual is the target value minus the predicted value. Boosting is defined for binary, nominal, and interval targets. The TREEBOOST procedure implements the algorithms described in "A Gradient Boosting Machine" and "Stochastic Gradient Boosting" by Jerome Friedman.

Like decision trees, boosting makes no assumptions about the distribution of the data. For an interval input, the model depends only on the ranks of the values. For an interval target, the influence of an extreme value theory depends on the loss function. The TREEBOOST procedure offers a Huber M-estimate loss that reduces the influence of extreme target values. Boosting is less prone to overfit the data than a single decision tree, and if a decision tree fits the data fairly well, then boosting often improves the fit.

Also refer to Recursive Partitioning Procedures in the Appendix for more details.

---

### Syntax: TREEBOOST Procedure

```
PROC TREEBOOST <option(s)>;
```

```

DECISION DECDATA= SAS-data-set<option(s)> ;
FREQ variable;
INPUT variables</option(s)>;
TARGET variable</option(s)>;
PERFORMANCE <option(s)>;
ASSESS <option(s)>;
CODE <option(s)>;
IMPORTANCE <option(s)>;
MAKEMACRO NTREES= macname;
SAVE <option(s)>;
SCORE <option(s)>;
SUBSERIES subseries;

```

The following table summarizes the function of each statement (other than the PROC statement) in the TREEBOOST procedure:

Statement	Description
DECISION	Specify profits and prior probabilities
FREQ	Specify a frequency variable
INPUT	Specify input variables with common options
TARGET	Specify the target variable
PERFORMANCE	Specify memory size and where to locate data
ASSESS	Evaluate subtrees and declare beginning of results
CODE	Generate SAS DATA step code for scoring new cases
IMPORTANCE	Modify variable values to infer its importance
MAKEMACRO	Define a macro variable
SAVE	Output data sets containing model results
SCORE	Use the model to make predictions on new data
SUBSERIES	Specify how many iterations in the series to use

Other than the SUBSERIES statement, these statements are described in the Recursive Partitioning Procedures section in the Appendix.

---

## PROC TREEBOOST Statement

```

PROC TREEBOOST <option(s)>;

```

The following table lists the options available in the PROC TREEBOOST statement for the tree-boosting tasks. The options HUBER, ITERATIONS, SHRINKAGE, TRAINN, and TRAINPROPORTION are unique to the tree boosting task and are described below. The other options are described in the Recursive Partitioning Procedures section in the Appendix.

Option	Description	Default
CATEGORICALBINS=	number of preliminary bins for categorical inputs	30
DATA=	training data set	
EVENT=	distinguished target category	smallest
EXHAUSTIVE=	maximum splits in complete enumeration	5000
HUBER=	threshold in Huber M-regression	NO
INMODEL=	data set containing model information	
INTERVALBINS=	number of preliminary bins for interval input values	100
INTERVALDECIMALS=	decimal precision of interval split point	MAX
ITERATIONS=	number of terms in the tree-boosting series	50
LEAFFRACTION=	LEAFSIZE as fraction of training data	0.001
LEAFSIZE=	minimum number of observations in a branch	
MAXBRANCH=	maximum number of branches from one node	2
MAXDEPTH=	maximum depth of a tree	2
MINCATSIZE=	observations needed for each category	5
MISSING=	policy for handling missing values	USEINSEARCH
SEED=	seed for pseudo random number generator	8976153
SHRINKAGE=	amount to reduce the prediction of each tree	0.2
SPLITSIZE=	minimum number of observations to split a node	10
TRAINN=	number of training observations for one tree	
TRAINPROPORTION=	proportion of data to use to train a tree	

**HUBER= $q$  | NO**

requests using the Huber M-regression loss function instead of square error loss with an interval target. The Huber loss function is less sensitive to extreme target values.  $Q$  must be between 0.6 and 1. A value close to one, such as 0.9, is reasonable.

HUBER=NO requests using square error loss instead of Huber loss. The HUBER option is ignored unless the target has an interval measurement level.

**Default:** HUBER=NO

**ITERATIONS=  $n$**

specifies the number of terms in the boosting series. For interval and binary targets, the number of iterations equals the trees. For a nominal target, a separate tree is created for each target category in each iteration, resulting in  $nJ$  trees, where  $J$  is the number of target values.  $N$  is an integer from 1 to 1000.

**Default:** In SAS 9.1.3, the default value of  $n$  is 50 for interval and binary targets and  $50/J$  for nominal targets.

**SEED=  $n$**

specifies the seed for generating random numbers. The TRAINN= and TRAINPROPORTION= option use random numbers to select a training sample at each iteration.  $N$  is a nonnegative integer. Set  $n$  to 0 to use the internal default.

**SHRINKAGE=  $p$**

specifies how much to reduce the prediction of each tree.  $P$  must be greater than 0 and less than or equal to 1.

**Default:** 0.2

**TRAINN=  $n$**

specifies how many observations to use to train each tree. The observations are counted without regard to the variable specified in the FREQ statement. Using less than all the available data might improve the generalization error. A different training sample is taken in each iteration. Trees that are trained in the same iteration have the same training data.  $N$  can be any positive integer. If  $n$  is greater than the number of observations in the data set specified in the DATA= option, then all the available data is used. The TRAINPROPORTION= option accepts a proportion instead of an absolute number to set the same quantity as the TRAINN= option. Specifying both TRAINN= and TRAINPROPORTION= is an error.

**Default:** By default, the TREEBOOST procedure uses all the available data in SAS 9.1.3.

**TRAINPROPORTION=  $p$**

specifies the proportion of training observations to train a tree with. Using less than all the available data might improve the generalization error. A different training sample is taken in each iteration. Trees trained in the same iteration have the same training data.  $P$  can be any number greater than 0 and at most 1. The TRAINN= option accepts an absolute number instead of a proportion to set the same quantity. Specifying both the TRAINN= and TRAINPROPORTION= is an error.

**Default:** The default value of  $p$  is 1.0 in SAS 9.1.3.

---

## ASSESS Statement

ASSESS <option(s)>;

The ASSESS statement accepts an assessment measure, uses it to evaluate each subseries of trees, and selects the best subseries as the model to use for prediction. The Recursive Partitioning Procedures section in the Appendix describes the statement and

its options. The following table lists the options available in the TREEBOOST procedure.

Option	Description
MEASURE=	specifies the assessment measure for selecting a model
NOPRIORS	ignores prior probabilities in model selection
PRIORS	incorporates prior probabilities in model selection
VALIDATA=	specifies a validation data set
NOVALIDATA	terminates a previous VALIDATA= option

---

## CODE Statement

**CODE** <option(s)>;

The CODE statement generates SAS DATA step code that mimics the computations done by the SCORE statement. The Recursive Partitioning Procedures section in the Appendix describes the statement and its options. The following table lists the options available in the TREEBOOST procedure.

Option	Description
CATALOG=	catalog to contain score code
FILE=	file to contain score code
FORMAT=	default format for interval values
GROUP=	specifies prefix for array names
LINESIZE   LS	line size for generated code
NORESIDUAL	requests score code to not compute residuals
RESIDUAL	requests score code to compute residuals

---

## DECISION Statement

**DECISION DECADATA=** *SAS-data-set* <option(s)> ;

The DECISION statement specifies decision functions and prior probabilities for categorical targets. The Recursive Partitioning Procedures section in the Appendix describes the statement. The following table lists the options available in the DECISION statement in the TREEBOOST procedure.

<b>Option</b>	<b>Description</b>
COST=	constant costs
DECDATA=	data set with decisions and prior probabilities
DECVAR=	decision variables in a DECDATA= data set
PRIORVAR=	variable of prior probabilities in DECDATA= data set

---

## **FREQ Statement**

**FREQ** *variable*;

The FREQ statement identifies a variable that contains the frequency of occurrence of each observation.

---

## **IMPORTANCE Statement**

**IMPORTANCE** *<option(s)>*;

The IMPORTANCE statement implements an observation-based approach to evaluate the importance of a variable or of a pair of variables to the predictions of the model. The Recursive Partitioning Procedures section in the Appendix describes the statement. The following table lists the options available in the TREEBOOST procedure.

<b>Option</b>	<b>Description</b>
DATA=	input data set
N2WAY=	encodes pairs of variables to evaluate
NVAR=	number of variables to evaluate alone
OUT=	output data set of predictions
OUTFIT=	output data set of goodness-of-fit statistics
VAR=	list of variables and pairs to evaluate

---

## **INPUT Statement**

**INPUT** *variables* *</option(s)>*;

The INPUT statement names input variables with common options. The INPUT statement can be repeated. The Recursive Partitioning Procedures section in the

Appendix describes the statement. The following table lists the options available for the tree-boosting task.

Option	Description
INTERVALDECIMALS=	decimal precision of interval split point
LEVEL=	measurement level
MAXBRANCHES=	maximum number of branches from one node
MINCATSIZE=	observations needed for each category
MISSING=	policy for handling missing values
ORDER=	order of values of ORDINAL inputs
SPLITATDATUM=	split interval input at a data point
SPLITBETWEEN=	split interval input between data points

---

## MAKEMACRO Statement

**MAKEMACRO** *ITERATIONS*=*macname*;

The MAKEMACRO statement specifies the name of a macro variable to contain a statistic of the model. The ITERATIONS= option specifies the name of a macro variable to contain the number of terms in the current subseries. For binary and interval targets, the number of iterations is the number of trees.

If the MAKEMACRO statement appears before a model is trained, the procedure trains a model before executing the statement. Subsequent initialization statements are prohibited.

---

## PERFORMANCE Statement

**PERFORMANCE** <*option(s)*>;

The PERFORMANCE statement specifies options affecting the speed of computations with little or no impact on the results. The Recursive Partitioning Procedures section in the Appendix describes the statement. The following table lists the options available for the tree-boosting task.

Option	Description
MEMSIZE=	size of RAM available for scratch work
NODESIZE=	sufficient size for within-node sample
WORKDATALOCATION=	where to copy training data (RAM, DISK, or no copy)

---

## SAVE Statement

**SAVE** *<option(s)>*;

The SAVE statement outputs model information into SAS data sets. The FIT= option is unique to the TREEBOOST procedure. The Recursive Partitioning Procedures section in the Appendix describes the other options. The following table lists the options available for the TREEBOOST procedure.

Option	Description
FIT=	fit statistics of each subseries
IMPORTANCE=	importance of variable based on splits
MODEL=	boosting model
NODESTATS=	node statistics for each node in each tree
RULES=	splitting rules

### **FIT=SAS-data-set**

specifies a data set to contain fit statistics for each iteration. The data set contains one observation for each iteration, and includes all the variables in the OUTFIT= data set option to the SAVE statement. The iteration number is in variable `_ITERATION_`.

---

## SCORE Statement

**SCORE** *<option(s)>*;

The SCORE statement reads a data set containing the input variables used by the model and outputs a data set containing the original variables plus new variables containing predictions, residuals, and decisions. The SCORE statement can be repeated. The Recursive Partitioning Procedures section in the Appendix describes the statement. The following table lists the options available for the TREEBOOST procedure.

Option	Description
DATA=	input data set
NOPREDICTION	suppress output of predictions
OUT=	output data set of predictions
OUTFIT=	output data set of goodness-of-fit statistics
PREDICTION	output predictions (default)
ROLE=	data set role

---

## SUBSERIES Statement

**SUBSERIES** *BEST* | *LONGEST* | *ITERATIONS= i* ;

The SUBSERIES statement specifies how many iterations in the series to use in the model. For a binary or interval target, the number of iterations is the number of trees. For a nominal target with  $k$  categories,  $k > 2$ , each iteration contains  $k$  trees.

**BEST**

selects the smallest subseries with the best assessment value.

**LONGEST**

selects the complete series.

**ITERATIONS= *i***

selects the subseries containing  $i$  iterations.

---

## TARGET Statement

**TARGET** *variable* < /*LEVEL = level* >;

The TARGET statement names the variable the model tries to predict. The level of measurement can be INTERVAL or NOMINAL. The default is INTERVAL for a numerical target variable, NOMINAL for character variable. Use NOMINAL to specify a binary target.

---

## Details: TREEBOOST Procedure

---

### FIT= Output Data Set

The FIT= option in the SAVE statement specifies a data set to contain fit statistics for all subseries of trees.